Proceedings Paper:
Escolano, Francisco, Curado, Manuel, Lozano, Miguel Angel et al. (1 more author) (2016) Dirichlet Graph Densifiers. In: Structural, Syntactic, and Statistical Pattern Recognition - Joint IAPR International Workshop, S+SSPR 2016, Mérida, Mexico, November 29 - December 2, 2016, Proceedings. Lecture Notes in Computer Science . Springer , pp. 185-195.

https://doi.org/10.1007/978-3-319-49055-7_17

Dirichlet Graph Densifiers

Francisco Escolano, Manuel Curado, Miguel A. Lozano, and Edwin R. Hancook

Department of Computer Science and AI, University of Alicante, 03690, Alicante (Spain)
{sco,mcurado,malozano}@dccia.ua.es
Department of Computer Science, University of York, York, YO10 5DD, UK
erh@york.ac.uk

Abstract. In this paper, we propose a graph densification method based on minimizing the combinatorial Dirichlet integral for the line graph. This method allows to estimate meaningful commute distances for mid-size graphs. It is fully bottom up and unsupervised, whereas anchor graphs, the most popular alternative, are top-down. Compared with anchor graphs, our method is very competitive (it is only outperformed for some choices of the parameters, namely the number of anchors). In addition, although it is not a spectral technique our method is spectrally well conditioned (spectral gap tends to be minimized). Finally, it does not rely on any pre-computation of cluster representatives.

Keywords: Graph densification, Dirichlet problems, Random walkers

1 Introduction

1.1 Motivation

kNN graphs have been widely used in graph-based learning, since they tend to capture the structure of the manifold where the data lie. However, it has been recently noted [1] that for a standard machine learning setting ($n \to \infty$, $k \approx \log n$ and large d, where n is the number of samples and d is their dimension) we have that kNN graphs result in a sparse, globally uninformative representation. In particular, a kNN-based estimation of the geodesics (for instance through the shortest paths as done in ISOMAP) diverges significantly unless we assign proper weights to the edges of the kNN. Finding such weights is a very difficult task as d increases. As a result, machine learning algorithms for graph-based embedding, clustering and label propagation tend to produce misleading results unless we are able of preserving the distributional information of the data in the graph-based representation. In this regard, recent experimental results with anchor graphs suggest a way to proceed. In [8][7][2], the predictive power of non-parametric regression rooted in the anchors/landmarks ensures a way of constructing very informative weighted kNN graphs from a reduced set of representatives (anchors). Since anchor graphs are bipartite (only *data-to-anchor* edges exist), this representation bridges the sparsity of the pattern space because a random walk traveling from node u to node v must reach one or more anchors in advance. In

other words, for a sufficient number of anchors it is then possible to find links between distant regions of the space. As a result, the problem of finding suitable weights for the graph is solved through kernel-based regression.

Data-to-anchor kNN graphs are computed from $m \ll n$ representatives (anchors) typically obtained through K-means clustering, in $O(dmnT+dmn)$, where $O(dmnT)$ is due to the T iterations of the K-means process. Since $m \ll m$, the process of constructing the $m \times m$ affinity matrix $W = Z\Lambda Z^T$, where $\Lambda = \text{diag}(Z^T 1)$ and Z is the data-to-anchor mapping matrix, is linear in n. As a byproduct of this construction, we have that the main r eigenvalue-eigenvector pairs associated with $M = \Lambda^{-1/2} Z^T Z \Lambda^{-1/2}$, which has also dimension $m \times m$, lead a compact solution for the spectral hashing problem [14] (see [9] for details). These eigenvectors-eigenvalues may also provide a meaningful estimation of the commute distances between the samples through the spectral expression of this distance [11].

Once considered the benefits of anchor graphs, their use is quite empirical since their foundations are poorly understood. For instance, the choice of the m representatives is quite open and heuristic. The K-means selection process outperforms the uniform selection because it approximates better the underlying distribution. More clever oracles for estimating not only the positions of the representatives but also their number would lead to interesting improvements. However, the developments of these oracles must be compatible with the underlying principle defining an anchor graph, namely *densification*. Densification refers to the process of increasing the number edges (or the weights) of an input graph so that the result preserves and even enforces the structural properties of the input graph. This is exactly what it anchors provide: given a sparse graph associated to a standard machine learning setting, they produce a more compact graph which is locally dense (specially around the anchors) and minimizes inter-class paths.

Graph densification is a principled study of how to significantly increase the number of edges of an input graph G so that the output, H, approximates G with respect to a given test function, for instance whether there exists a given cut. Existing approaches [4] pose the problem in terms of semidefinite programming SDP where a global function is optimized. These approaches have two main problems: a) the function to optimize is quite simple and does not impose the minimization of inter-class edges while maximizing intra-class edges, and b) since the number of unknowns is $O(n^2)$, i.e. all the possible edges, and SDP solvers are polynomial in the number of unknowns [10], only small-scale experiments can be performed. However, these approaches have inspired the densification solution proposed in this paper. Herein, instead of proposing an alternative oracle (top-down solution) we contribute with a method for grouping sparse edges so that densification can rely on similarity diffusion (bottom-up solution). Since our long-term scientific strategy is to find a meeting point between bottom-up and top-down densifiers, here we study to what extent we can approximate the performance of anchor graphs from the input sparse graph as a unique source of information.

1.2 Contributions

In this paper, we propose a bottom-up graph densification approach which commences by grouping edges through *return random walks* (Section 2). Return random walks (RRW) are designed to enforce intra-class edges while penalizing inter-class weights. Since our strategy is completely unsupervised, return random walks operate under the hypothesis that inter-class edges are rare events. Given input sparse graph G (typically resulting from a thresholded similarity matrix W), RRWs produce a probabilistic similarity matrix W_e. Then, high probability edges are assumed to drive the grouping process. To this end, we exploit the *random walker* [3] but in the edges space (Section 3). The random walker minimizes the Dirichlet integral, in this case that associated with the line graph of W_e: $Line_{W_e}$. Given a set of known edges (assumed to be the ones with maximal probability in W_e) we predict the remainder edges. The result is a locally-dense graph H that is suitable for computing commute distances. In our experiments (Section 4), we will compare our *Dirichlet densifier* with anchor graph as well as with existing non-spectral alternatives relying exclusivelly on kNN graphs.

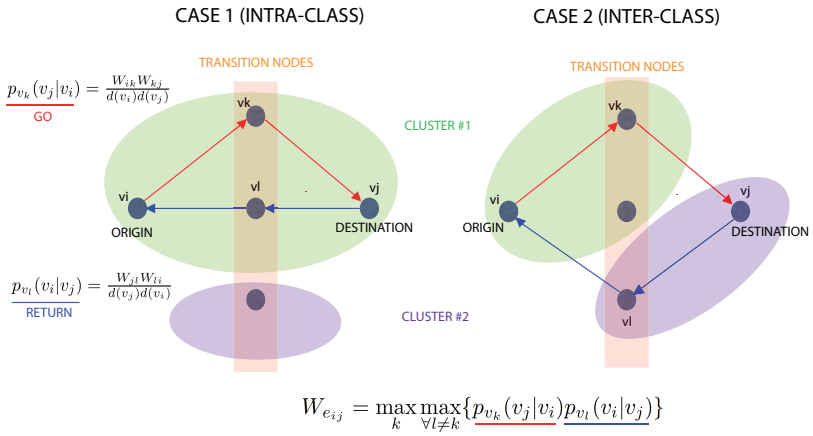

Fig. 1. Return random walks for reducing inter-class noise.

2 Return Random Walks

Given a set of points $\chi = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\} \subset \mathbb{R}^d$, we map the $\boldsymbol{x}_i$ to the vertices V of an undirected weighted graph $G(V, E, W)$. We have that V is the set of nodes where each v_i represents a data point x_i , $E \subseteq V \times V$ is the set of edges linking adjacent nodes. An edge $e = (i, j)$ with $i, j \in V$, exists if $W_{ij} > 0$ where $W_{ij} = e^{-\sigma||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}$, i.e. $W \in \mathbb{R}^{n \times n}$ is a weighted similarity matrix.

Design of We. Given W we produce a reweighted similarity matrix W_e by following this rationale: a) we explore the two-step random walks reaching a node v_j from v_i through any transition node v_k, b) *on return* from v_j to v_i we maximize the probability of returning through a different transition node $v_l \neq v_k$. For the first step (going from v_i to v_j through v_k) we have $p_{v_k}(v_j|v_i) = \frac{W_{ik}W_{kj}}{d(v_i)d(v_j)}$ as well as a *standard return* $p_{v_l}(v_i|v_j) = \frac{W_{jl}W_{li}}{d(v_j)d(v_i)}$. Standard return works pretty well if v_i and v_j belong to the same cluster (see Fig. 1-left). However, v_l (the transition node for returning) can be constrained so that $v_l \neq v_k$. In this way, travelling out of a class is penalized since the walker must choose a different path, which in turn is hard to find on average. Therefore, we obtain $W_{e_{ij}}$ from W_{ij} as follows:

$$W_{e_{ij}} = \max_k \max_{\forall l \neq k} \{p_{v_k}(v_j|v_i)p_{v_l}(v_i|v_j)\} \,, \tag{1}$$

i.e. for each possible transition node v_k we compute the probability of go and return (product of independent probabilities) through a different node v_l. We retain the maximum product of probabilities for each v_l referred to a given k and finally we retain the supremum of these maxima. As a result, when inter-class travels are frequent for a given $e = (i, j)$ (Fig. 1-right) its weight $W_{e_{ij}}$ is significantly reduced. Our working hypothesis is that the number of edges subject to this condition is small on average, since the number of inter-class edges tends to be small compared with the total number of edges. However, in realistic situations where patterns can be confused due either to their intrinsic similarity or to the use of an unproper similarity measure, this assumption leads to a significant decrease of many weights of W.

3 The Dirichlet Graph Densifier

3.1 The Line Graph

The graph densification problem can be posed as follows: given a graph $G = (V, E, W)$ infer another graph $H = (V, E', W')$ so that $|E'| \geq |E|$ in such a way that the bulk of the increment in the number of edges is constrained to intra-class edges (i.e. the number of inter-class edges is minimized). Therefore, the unknowns of the problem are the new edges to infer, not the vertices. In principle we have a $O(n^2)$ unknowns, where $n = |V|$, but working with all of them is infeasible. This motivates the selection of a small fraction of them (those with the highest values of $W_{e_{ij}}$) according to a given theshold γ_e. The counterintuitive fact that the smaller the fraction the better the accuracy is explained below and showed later in the experimental section. Concerning efficiency, the first impact of this choice is that only $|E''|$ edges, with $|E''| \ll |E|$ are considered for building a graph of edges, i.e. a *line graph* $Line_{W_e}$. Let A the $p \times n$ edge-node incidence matrix defined as follows:

$$A_{e_{ij}v_k} = \begin{cases} +1 \text{ if } i = k, \\ -1 \text{ if } j = k, \\ 0 \quad \text{otherwise}, \end{cases} \tag{2}$$

Then, the $C = AA^T - 2I_p$ is the adjacency matrix of an unweighted line graph, where: I_p is the $p \times p$ identity matrix, the nodes e_a are given by all the possible pairs of $r = |E''|$ edges with a common vertex according to A. The edges of C implement second-order interactions between nodes in the original graph from which A comes from. However, C is still unattributed (although conditioned by W_e). A proper weighting of for this graph is to use standard "go and return" random walks, i.e.

$$Line_{W_e}(e_a, e_b) = \sum_{k=1}^{r} p_{e_k}(e_b|e_a)p_{e_k}(e_a|e_b) \ , \tag{3}$$

i.e. return walks are not applied because they become too restrictive. Then, there is an edge in the line graph for every pair (e_a, e_b) with $Line_{W_e}(e_a, e_b) > 0$. We denote the set os edges of the line graph by E_{Line}

3.2 The Dirichlet Functional for the Line Graph

Given the line graph $Line_{W_e}$ with r nodes (now *edges*) many of them will be highly informative according to W_e and the application of Eq. 3. We retain a fraction of them (again, those with the largest values of W_e) according to a second threshold μ_e. This threshold must be set as smaller as possible since it defines the difference between the "known" and the "unknown". More precisely, W_e acts as a function $W_e : |E''| \rightarrow \mathbb{R}$ so that the larger its value the more trustable is a given edge as an stable or known edge in the original graph G. Unknown edges are assumed to have small values of W_e and this is why they are not selected, since the purpose of our method is to infer them.

This is a classical inference problem, now in the space of edges and completely unsupervised, which has been posed in terms of minimizing the disagreements between the weights of existing (*assumed* to be "known") edges and those of the "unknown" or inferred ones. In this regard, since unknown edges are typically neighbors of known ones, the minimization of this disagreement is naturally expressed in terms of finding an harmonic function. Harmonic functions $u(x)$ satisfy $\nabla^2 u = 0$ which in our discrete setting leads to the following property

$$u(e_a) = \frac{1}{d(e_a)} \sum_{(e_a,e_b)\in E_{Line}} Line_{W_e}(e_a, e_b)u(e_b) \ , \tag{4}$$

The harmonic function $u(.)$ is not unconstrained, since it is known for some values of the domain (the so called *boundary*). In our case, we set $u(e_a) = W_{e_a}$ for $e_a \in E_B$, referred to as *border nodes* since they are associated with assumed known edges. The harmonic function is unknown for $e_b \in E_I = E'' \sim E_B$ (the *inner nodes*). Then, finding an harmonic function given boundary values is called the *Dirichlet problem* and it is typically formulated in terms of minimizing the following integral

$$D[u] = \frac{1}{2}\int_{\Omega} |\nabla u|^2 d\Omega, \tag{5}$$

whose discrete version relies on the graph Laplacian [3] (in this case on the Laplacian of the line graph):

$$D_{Line}[u] = \frac{1}{2}(Au)^T R(Au) = \frac{1}{2}u^T \mathcal{L}_{Line} u$$
$$= \frac{1}{2} \sum_{(e_a,e_b)\in E_{Line}} Line_{W_e}(e_a, e_b)(u(e_a) - u(e_b))^2 \ , \quad (6)$$

where A' is the $|E"| \times |E_{Line}|$ incidence matrix, R is the $|E_{Line}| \times |E_{Line}|$ diagonal constitutive matrix containing all the weights of the edges in the line graph, and $\mathcal{L}_{Line} = D_{Line} - Line_{W_e}$ with $D_{Line} = diag(d(e_a) \dots d(e_{|E"|}))$ where $d(e_a) = \sum_{e_b \neq e_a} Line_{W_e}(e_a, e_b)$ is the diagonal degree matrix. Then, $\mathcal{L}_{Line}$ is the Laplacian of the line graph.

Given the Laplacian $\mathcal{L}_{Line}$ and the Dirichlet combinatorial integral D_{Line} we have that the nodes in the line graph are partitioned in two classes: "border" and "inner", i.e. $E" = E_B \cup E_I$. This partition leads to a reordering of the harmonic function $u = [u_B \ u_I]$ as well as the Dirichlet integral:

$$D\left[u_I\right] = \tfrac{1}{2}\left[u_B^T \ u_I^T\right] \begin{bmatrix} L_B & K \\ K^T & L_I \end{bmatrix} \begin{bmatrix} u_B \\ u_I \end{bmatrix} \quad (7)$$

where $D\left[u_I\right] = \frac{1}{2}(u_B^T L_B u_B + 2u_I^T K^T U_B + u_I^T L_I u_I)$ and differentiating w.r.t. u_I leads to solve a linear system which relates u_I with u_B:

$$L_I u_I = -K^T u_B \ . \quad (8)$$

Then, let $s \in [0, 1]$ be a label indicating to what extend a given node of the line graph (an edge in the original graph) is relevant. We define a potential function $Q : E_B \to [0, 1]$ so that for a known node $e_a \in E_B$ we assign a label s, i.e. $Q(e_a) = s$. This leads to declaring the following vector for each label:

$$m_a^s = \begin{cases} \frac{W_{e_a}}{\max_{e_b \in E"}\{W_{e_b}\}} & \text{if } Q(e_a) = s, \\ 0 & \text{if } Q(e_a) \neq s \end{cases} \ . \quad (9)$$

Finally, the linear system is posed in terms of how the known labels do predict the unknown ones, placed in the vector u, as follows:

$$L_I u^s = -K^T m^s \ . \quad (10)$$

If we consider simultaneously all labels instead of a single one, we have

$$L_I U = -K^T M \ \Rightarrow U = (-K^T M) L_I^{-1} \ , \quad (11)$$

where U is a vector of $|E_I|$ rows (one per unknown/inner edge, *known solved*) and M has $|E_B|$ rows and columns. Then, let U_b be the $b-$th row, i.e. the weight of a previously unknown edge e_b. Since there is a bijective correspondence between the nodes in the line graph (some of them are denoted by e_a since they are

known, and the remainder are denoted by e_b) and the edges in the original graph $G = (V, E, W)$, then we have that e_k corresponds to edge $(i, j) \in E$. However, since its weight has potentially changed after solving the linear system, we adopt the following densification criterion (labeling) for creating the graph $H = (V, E, W')$:

$$H_{ij} = \begin{cases} \max_{e_k \in U} U_k & \text{if } e_k \in E_I \\ M_{ij} & \text{if } e_k \in E_B, \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

In this way, the edges E' of the dense graphs are given by $H_{ij} > 0$.

Perc. of Known labels	**Perc. of Edges Selected**									
	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%
5%	0.42	0.34	0.31	0.25	0.19	0.18	0.13	0.10	0.10	0.10
10%	0.49	0.42	0.41	0.36	0.31	0.31	0.23	0.20	0.16	0.16
15%	0.49	0.46	0.48	0.46	0.40	0.34	0.28	0.24	0.19	0.18
20%	0.51	0.50	0.53	0.40	0.38	0.34	0.30	0.28	0.24	0.19
25%	0.53	0.54	0.49	0.47	0.39	0.30	0.32	0.26	0.24	0.17
30%	0.55	0.53	0.48	0.43	0.34	0.34	0.29	0.20	0.24	0.23
35%	0.59	0.58	0.51	0.37	0.37	0.26	0.20	0.21	0.18	0.17
40%	0.56	0.52	0.47	0.36	0.27	0.21	0.23	0.18	0.17	0.14
45%	0.56	0.54	0.44	0.32	0.25	0.23	0.18	0.18	0.19	0.20
50%	0.60	0.55	0.39	0.35	0.22	0.18	0.16	0.19	0.16	0.14

Table 1. Dirichlet densifier: Accuracy for the reduced NIST database

4 Experiments and Conclusions

In our experiments we use a reduced version of the NIST digits database: $n = 200$ (20 samples per class) and proceed to estimate commute distances. In all cases, given a similarity matrix we use the $O(nlogn)$ randomized algorithm proposed in [12] . We explore the behavior of the proposed Dirichlet densifier for different values of γ_e, the threshold leading to preserve different fractions of the leading edges (the ones with the highest values in W_e: from 5% to 50%. Concerning the theshold μ_e controlling the fraction of leading nodes in the line graph assumed to be "known" (i.e. border data in the terminology of Dirichlet problems), we have explored the same range: from 5% to 50% (see Table 1 where we show the accuracies corresponding to each of the 100 experiments performed. A first important conclusion is that the best clustering accuracy (w.r.t. the ground truth) is obtained when the fraction of retained edges for constructing the line graph is minimal. Although the removed edges cannot be reconstructed after solving the Dirichlet equation, i.e. we bound significantly the level of densification, reducing the fraction of retained edges reduces significantly the inter-class noise. We show this effect in Figs. 2 (e)-(f). For instance, a fraction of 5% (c) produces a better

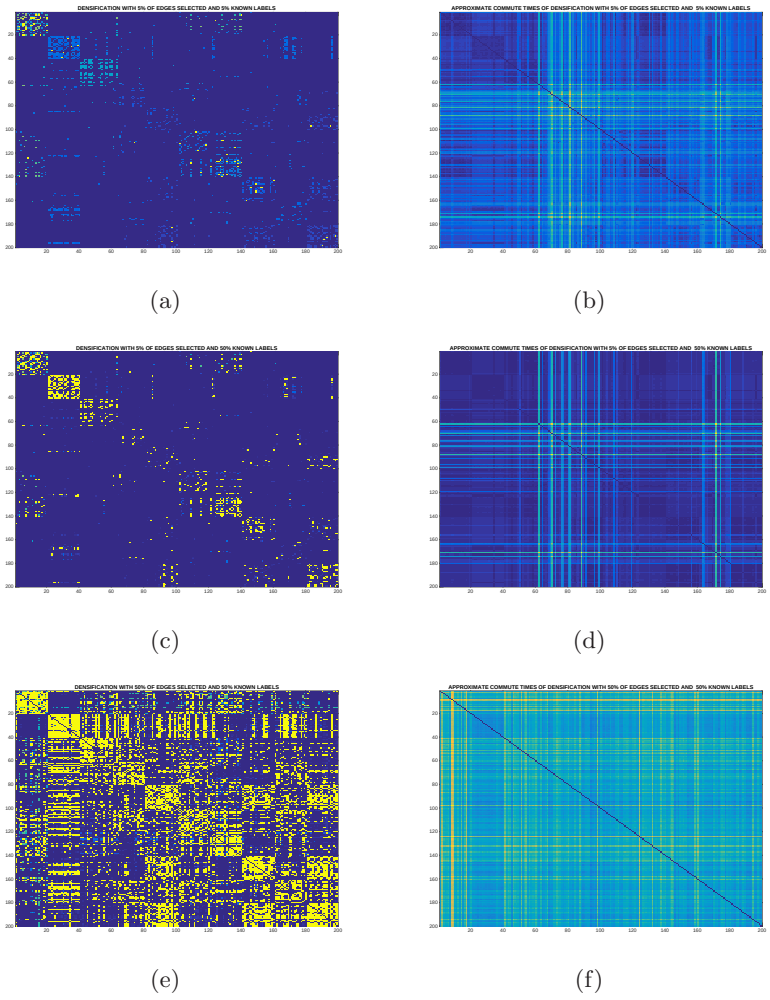

Fig. 2. Densification result and its associate Approximate commute times (ACT) matrix for different fractions of known labels $|E_B|$ and leading edges $|E"|$: (a) Densification with $|E_B| = 5\%$, $|E"| = 5\%$, (b) corresponding ACT , (c) Densification with $|E_B| = 50\%$, $|E"| = 5\%$, (d) corresponding ACT , (e) Densification with $|E_B| = 50\%$, $|E"| = 50\%$, (f) corresponding ACT.

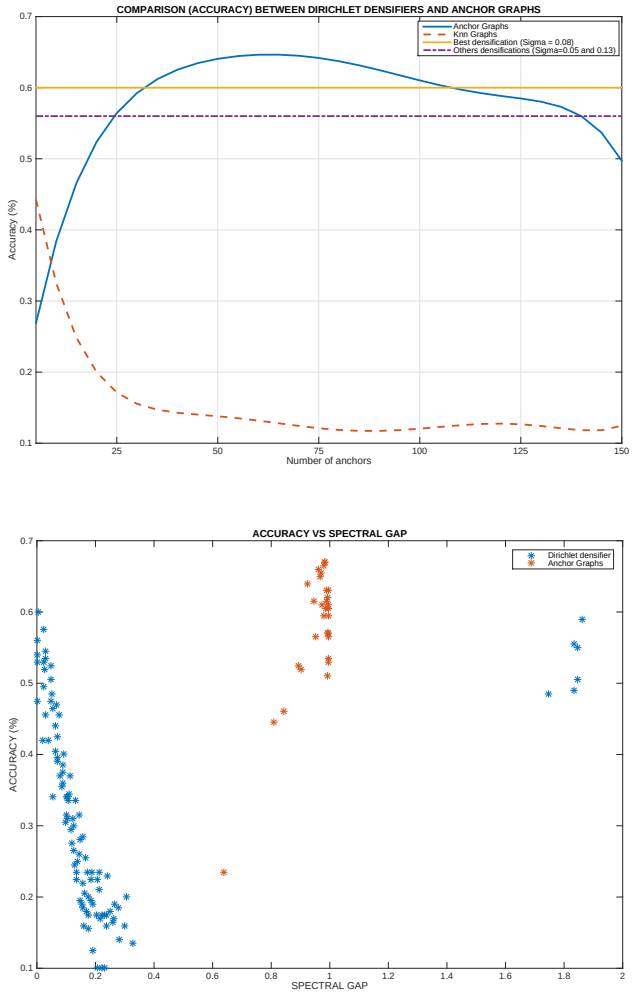

Fig. 3. Top: Accuracy of Anchors Graphs, kNN Graphs and Dirichlet densifiers. Dirichlet densifiers do not depend on the number of anchors and are completely unsupervised. Bottom: Accuracy vs Spectral Gap.

approximation of the commute distance (d) w.r.t. retaining 50% of the edges to build the line graph. The commute distances after retaining 50% are meaningless (f) despite the obtained graph is denser. In all cases, the error assumed when approximating the commute times matrix is $\epsilon = 0.25$.
In a second experiment, we compare the commute distances obtained with the optimal Dirichlet densifier (fraction of retained leading edges $|E"| = 5\%|$ and fraction known labels $|E_B| = 50\%$) with different settings for the anchor graphs. Concerning anchor graphs, in all cases we set $\sigma = 0.08$ for constructing the Gaussian graphs from the raw input data. In our Dirichlet approach we use the same setting. This provides the best result in the range $\sigma \in [0.05, 0.13]$. In Fig. 3-Left we show how the accuracy evolves while increasing the number of (anchors) m: from 5 to 150. The performance of anchor graphs increases with m but degrades after reaching the peak at $m = 70$ (accuracy 0.67). This peak is due to the fact that anchor graphs tend to reduce the amount of inter-class noise. However, this often leads to poor densification. On the other hand, Dirichlet densifiers they are completely unsupervised and do not rely on anchor computation. Their performance is constant w.r.t. m and the best accuracy is 0.60. We outperform anchor graphs for $m < 35$ and $m > 105$ and in the range $m \in [35, 105]$ our best accuracy is very close to the anchor graph's performance. Regarding existing approaches that compute commute distances from standard weighted kNN graphs [6][5] we outperform them for any choice of m, since their performace degrades very fast with m due to the intrinsic inter-class noise arising in realistic databases.

Finally, we reconcile our results, and those of the anchor graphs with the von Luxburg and Radl's fundamental bounds. In principle, commute distances cannot be properly estimated from large graphs [13]. However, in this paper we show that both anchor graphs and Dirichlet densifiers provide meaningful commute times. It is well known that this can be done insofar the spectral gap is close to zero or the minimal degree is close to the unit. Dirichlet densifiers provide spectral gaps close to zero (see Fig. 3-Right) for low fractions of leading edges, but the accuracy degrades linearly when the spectral gap increases. This means that the spectral gap is negatively correlated with increasing fractions of inter-class noise. This noise arises when the densification level increases since Dirichlet densifiers are not still able of confining densification to intra-class links. Concerning anchor graphs, their spectral gap is close to the unit since the degree also the unit (double-stochastic matrices) and they outperform Dirichlet densifiers to some extent at the cost of computing anchors and finding the best number of them.

To conclude, we have contributed with a novel method for transforming input graphs into denser versions which are more suitable for estimating meaningful commute distances in large graphs.

References

1. Alamgir, M., von Luxburg, U.: Shortest path distance in random k-nearest neighbor graphs. In: Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012. (2012)

2. Cai, D., Chen, X.: Large scale spectral clustering via landmark-based sparse representation. IEEE Trans. Cybernetics **45**(8) (2015) 1669–1680
3. Grady, L.: Random walks for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **28**(11) (2006) 1768–1783
4. Hardt, M., Srivastava, N., Tulsiani, M.: Graph densification. In: Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012. (2012) 380–392
5. Khoa, N.L.D., Chawla, S.: Large scale spectral clustering using approximate commute time embedding. CoRR **abs/1111.4541** (2011)
6. Khoa, N.L.D., Chawla, S.: Large Scale Spectral Clustering Using Resistance Distance and Spielman-Teng Solvers. In: Discovery Science: 15th International Conference, DS 2012, Lyon, France, October 29-31, 2012. Proceedings. Springer Berlin Heidelberg, Berlin, Heidelberg (2012) 7–21
7. Liu, W., He, J., Chang, S.: Large graph construction for scalable semi-supervised learning. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel. (2010) 679–686
8. Liu, W., Wang, J., Chang, S.: Robust and scalable graph-based semisupervised learning. Proceedings of the IEEE **100**(9) (2012) 2624–2638
9. Liu, W., Wang, J., Kumar, S., Chang, S.: Hashing with graphs. In: Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011. (2011) 1–8
10. q. Luo, Z., k. Ma, W., c. So, A.M., Ye, Y., Zhang, S.: Semidefinite relaxation of quadratic optimization problems. IEEE Signal Processing Magazine **27**(3) (2010) 20–34
11. Qiu, H., Hancock, E.R.: Clustering and embedding using commute times. IEEE Trans. Pattern Anal. Mach. Intell. **29**(11) (2007) 1873–1890
12. Spielman, D.A., Srivastava, N.: Graph sparsification by effective resistances. SIAM J. Comput. **40**(6) (2011) 1913–1926
13. von Luxburg, U., Radl, A., Hein, M.: Hitting and commute times in large random neighborhood graphs. Journal of Machine Learning Research **15**(1) (2014) 1751–1798
14. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008. (2008) 1753–1760